CERTIFYING SOFTWARE FIT FOR PURPOSE

Graeme Parkin and Peter Harris,
National Physical Laboratory
 21 October 2005

1 Introduction

The use of software in measurement systems has dramatically increased over the last few years, making devices easier to use, more reliable and more accurate. However the hidden complexity within the software is a potential source of undetected errors. Since it is hard to quantify the reliability or quality of such software, two questions arise:

- As a user of such a system, how can I be assured that the software is of sufficient quality to justify its use?
- As a supplier of such software, what validation techniques should I use, and how can I assure my users of the quality of the resulting software?

A means to certify that software is fit for purpose is required by both users and suppliers of measurement systems. It is not possible to test software exhaustively. There are many examples reported in the public domain of errors in software that have been very costly, either in money or life. For example, the Ariane 5 launcher ended in failure, the launcher veered off its flight path, broke up and exploded costing $370 million, due to a wrong conversion of a 64 bit value[1]. So even when best practice has been applied software can still have bugs. There are many possible techniques that can be applied in the development of software to reduce the number of errors. However the application of these techniques costs both time and money with diminishing returns.

An approach is described which determines which techniques should be used to produce software fit for purpose. This is illustrated by an example. It is also explained why instrument manufacturers are interested in this work for certifying products for safety-critical applications.

2 A solution

A risk analysis approach is taken to determine the techniques to be applied in the development of software which is fit-for-purpose. The risk analysis is based on three parameters, criticality of usage, complexity of processing and complexity of control, to which values are assigned. Each parameter can take one of four values. Criticality of usage values are one of critical, business critical, potentially safety-critical and safety-critical. Complexity of processing values are one of very simple, simple, moderate and complex. Complexity of control values are one of very simple, simple, moderate and complex. A further consideration is any legal obligations that may have to be met. Having assigned values to the risk parameters a Measurement Software Level (MSL) is determined based on Table 1. Having calculated a MSL, Table 2 is used to determine the techniques to develop the software so that it is fit-for-purpose. The Guide assumes a quality system is in place e.g ISO 9000 series of standards[2].

3 Application

In a recent application of the approach it was required to produce reference software for the calculation of surface texture parameters based on a profile[3] and be able to read profile data in SMD format[4]. The software was also required to work across platforms and give the same results on each. An example of a parameter is shown in Figure 1. Figure 2 shows briefly the derivation of the MSL, the techniques to be used to meet that MSL and the tools used. Other tools used were an IDE (BlueJ 2.0.3), component testing (JUnit 3.8.1) and a Java-based build tool (Ant 1.6.2).

---

[1] N° 33-1996: Ariane 501 - Presentation of Inquiry Board report.

[2] ISO 9001 2000: Quality management systems -- Requirements, ISO IEC 90003 2004: Software engineering - Guidelines for the application of ISO 9001:2000 to computer software.

[3] ISO 4287 Geometrical Product Specifications (GPS) -- Surface texture: Profile method -- Terms, definitions and surface texture parameters. 1997.

[4] ISO 5436-2 Geometrical Product Specifications (GPS) -- Surface texture: Profile method; Measurement standards -- Part 2: Software measurement standards. 2001.

4 The guide

The process outlined in the previous sections is much more fully described in Best Practice Guide No1, Validation of Software in Measurement Systems[5]. The guide has been designed to be used as the basis of certification services mainly with auditable checklists.

Criticality of usage	Complexity of Processing	Impact of complexity of control			
		Very simple	Simple	Moderate	Complex
Critical	Very simple	0	0	1	2
	Simple	0	1	1	2
	Moderate	1	1	2	2
	Complex	2	2	2	2
Business Critical	Very simple	0	1	1	2
	Simple	1	1	2	2
	Moderate	1	2	2	2
	Complex	2	2	2	3
Potentially life-critical	Very simple	1	1	2	2
	Simple	1	2	2	3
	Moderate	2	2	3	3
	Complex	2	3	3	3
Life-critical	Very simple	2	2	2	3
	Simple	2	2	2	3
	Moderate	2	2	3	4
	Complex	3	3	4	4

Table 1 Measurement Software Level as function of risk factors (see Guide for further details)

Furthermore the guide, when used for safety-critical software, assists compliance with the international standard for functional safety IEC 61508. The guide is to be used as input to determining a means to certify products to IEC 61508 by the 61508 Association[6] which was set up by instrument manufacturers in the UK. Currently the guide is being used to evaluate the software in alarm annunciators for Evaluation International[7].

5 Development of the guide

The guide was designed to provide advice which would satisfy a range of standards including: ISO/IEC 17025[8], Legal metrology[9], IEC 601-1-4[10], IEC 61508[11] and DO-178B[12]. The techniques

[5] Software Support for Metrology, Best Practice Guide No. 1, Validation of Software in Measurement Systems Brian Wichmann, Graeme Parkin and Robin Barker March 2004, Version 2.1, http://www.npl.co.uk/ssfm/download/documents/ssfmbpg1.pdf (freely available).

[6] http://www.61508.org.uk/

[7] http://www.evaluation-international.com/

[8] ISO/IEC 17025: 2005. General requirements for the competence of testing and calibration laboratories.

[9] WELMEC 2.3 Guide for examining software (Non-automatic weighing instruments), January 1995. WELMEC 7.1 Software requirements on the basis of the measuring instruments directive, January 2000. Both available at http://www.welmec.org/pubs.asp.

[10] IEC 601-1-4 Medical electrical equipment – Part 1: General requirements for safety 4: Collateral standard: Programmable electrical medical systems.

[11] IEC 61508: Parts 1-7, Functional safety of electrical/electronic/programmable electronic (E/E/PE) safety-related systems.

mentioned in the guide have been selected based on industry acceptance, tool support and ease of being audited. The guide has been reviewed, and their comments taken into account by persons in the following application areas of nuclear, medical, safety-critical and certification.

Ref.	*Recommended Technique*	*Measurement Software Level*			
		1	2	3	4
12.2	Review of informal specification	Yes	Yes		
12.3	Software inspection of specification		Yes	Yes	
12.4	Mathematical specification	Yes	Yes	Yes	Yes[13]
12.5	Formal specification				Yes[13]
12.6	Static analysis		Yes	Yes	Yes[13]
12.6	Boundary value analysis		Yes	Yes	
12.7	Defensive programming	Yes	Yes		
12.8	Code review	Yes	Yes		
12.9	Numerical stability		Yes	Yes	Yes[13]
12.10	Microprocessor qualification				Yes[13]
12.11	Verification testing			Yes	Yes[13]
12.12	Statistical testing		Yes	Yes	
12.13	Structural testing	Yes			
12.13	Statement testing		Yes	Yes	
12.13	Branch testing			Yes	Yes[13]
12.13	Boundary value testing		Yes	Yes	Yes[13]
12.13	Modified Condition/Decision testing				Yes[13]
12.15	Accredited testing		Yes		
12.16	System-level testing	Yes	Yes		
12.17	Stress testing		Yes	Yes	
12.18	Numerical reference results	Yes	Yes	Yes[13]	Yes[13]
12.19	Back-to-back testing		Yes	Yes	
12.20	Source code with executable				Yes[13]

Table 2 Recommended Techniques (see the guide for further details)

$$RSm = \frac{1}{m}\sum_{i=1}^{m} Xs_i$$

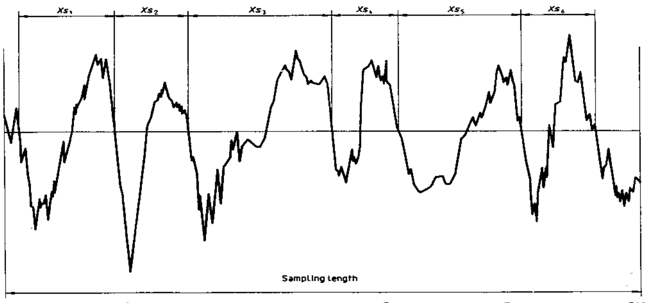

Figure 1 Spacing parameter RSm for a roughness profile

[12] DO-178B Software Considerations in Airborne Systems and Equipment Certification. Issued in the USA by the Requirements and Technical Concepts for Aviation (document RTCA SC167/DO-178B) and in Europe by the European Organization for Civil Aviation Electronics (EUROCAE document ED-12B). December 1992.
[13] These are still suggestions for MSL4 or, in the case of MSL3 are to be used if no alternative.

Risk analysis
- No legal requirements
- Business critical
- Simple complexity of control
- Moderate complexity of processing

(plus other issues like ease of testing etc.)

Measurement Software Level = 2

Ref.	*Recommended Technique*	*MSL 2*	*Used*	*How this is met*
12.2	Review of informal specification	Yes	Yes	-
12.3	Software inspection of specification	Yes	No	Based on international standard
12.4	Mathematical specification	Yes	Yes	MATLAB 7.0
12.5	Formal specification			Not applicable
12.6	Static analysis	Yes	Yes	Java compiler 1.4.2_4, Checkstyle 3.3
12.6	Boundary value analysis	Yes	Yes	-
12.7	Defensive programming	Yes	Yes	-
12.8	Code review	Yes	Yes	Checkstyle 3.3
12.9	Numerical stability	Yes	Yes	-
12.10	Microprocessor qualification			Not applicable
12.11	Verification testing			Not applicable
12.12	Statistical testing	Yes	No	-
12.13	Structural testing			Not applicable
12.13	Statement testing	Yes	Yes	Clover 1.3_02
12.13	Branch testing		Yes	Clover 1.3_02
12.13	Boundary value testing	Yes	Yes	-
12.13	Modified Condition/Decision testing			Not applicable
12.15	Accredited testing	Yes	No	Not applicable
12.16	System-level testing	Yes	Yes	-
12.17	Stress testing	Yes	Yes	Tested for large data sets
12.18	Numerical reference results	Yes	No	-
12.19	Back-to-back testing	Yes	Yes	Against MATLAB specifications
12.20	Source code with executable			Not applicable

Figure 2 Shows derivation of MSL and techniques used for the surface texture reference software

6 Summary

A means to certify software so that is fit for purpose has been briefly described. A service to certify software using the guide is being set up. Further work includes getting the guide more widely accepted, possibly through standardisation and developing guides on the use, application and evaluation of software development tools e,g, code coverage tools.